Foundations of Informatics: a Bridging Course

Part C: Context-Free Languages
Lesson 5: The Emptiness Problem for Context-Free Languages

Thomas Noll
Software Modeling and Verification Group
RWTH Aachen University

https://moves.rwth-aachen.de/people/noll/

Software Modeling and Verification Chair

RWTH AACHEN UNIVERSITY

The Emptiness Problem

Emptiness Problem for CFL

Given CFG $G = \langle N, \Sigma, P, S \rangle$, decide whether $L(G) = \emptyset$ or not.

The Emptiness Problem

Emptiness Problem for CFL

Given CFG $G = \langle N, \Sigma, P, S \rangle$, decide whether $L(G) = \emptyset$ or not.

- Important problem with many applications
  - consistency of context-free language definitions
  - correctness properties of recursive programs
  - ...
- For regular languages this was easy: check in the corresponding DFA whether some final state is reachable from the initial state.
- Here: test whether start symbol is productive, i.e., whether it generates a terminal word

The Emptiness Test

Algorithm (Emptiness Test)

Input: $G = \langle N, \Sigma, P, S \rangle$

Question: $L(G) = \emptyset$?

Procedure: *mark every* $a \in \Sigma$ *as productive*;

```
repeat
  if there is A → α ∈ P such that all symbols in α productive then
    mark A as productive;
  end;
until no further productive symbols found;
```

Output: *"no" if* S *productive, otherwise "yes"*

The Emptiness Test

Algorithm (Emptiness Test)

Input: $G = \langle N, \Sigma, P, S\rangle$

Question: $L(G) = \emptyset$?

Procedure: *mark every* $a \in \Sigma$ *as productive*;

```
repeat
  if there is A → α ∈ P such that all symbols in α productive then
    mark A as productive;
  end;
until no further productive symbols found;
```

Output: *"no" if S productive, otherwise "yes"*

Example

$$
\begin{aligned}
G:\quad S &\to AB \mid CA\\
A &\to a\\
B &\to BC \mid AB\\
C &\to aB \mid b
\end{aligned}
$$

The Emptiness Test

Algorithm (Emptiness Test)

Input: $G = \langle N, \Sigma, P, S\rangle$

Question: $L(G) = \emptyset$?

Procedure: *mark every* $a \in \Sigma$ *as productive*;

```
repeat
  if there is A → α ∈ P such that all symbols in α productive then
    mark A as productive;
  end;
until no further productive symbols found;
```

Output: *"no" if S productive, otherwise "yes"*

Example

$$
\begin{aligned}
G:\quad S &\to AB \mid CA \\
A &\to a \\
B &\to BC \mid AB \\
C &\to aB \mid b
\end{aligned}
$$

1. Initialisation

The Emptiness Test

Algorithm (Emptiness Test)

Input: $G = \langle N, \Sigma, P, S \rangle$
Question: $L(G) = \emptyset$?
Procedure: mark every $a \in \Sigma$ as productive;
    `repeat`
        `if` there is $A \to \alpha \in P$ such that all symbols in α productive `then`
            mark A as productive;
        `end`;
    `until` no further productive symbols found;
Output: "no" if S productive, otherwise "yes"

Example

$$
\begin{aligned}
G: \quad S &\to AB \mid CA \\
A &\to a \\
B &\to BC \mid AB \\
C &\to aB \mid b
\end{aligned}
$$

1. Initialisation
2. 1st iteration

The Emptiness Test

Algorithm (Emptiness Test)

Input: $G = \langle N, \Sigma, P, S \rangle$

Question: $L(G) = \emptyset$?

Procedure: mark every $a \in \Sigma$ as productive;

```
repeat
   if there is A → α ∈ P such that all symbols in α productive then
      mark A as productive;
   end;
until no further productive symbols found;
```

Output: "no" if S productive, otherwise "yes"

Example

$$
\begin{aligned}
G: \quad S &\to AB \mid CA \\
A &\to a \\
B &\to BC \mid AB \\
C &\to aB \mid b
\end{aligned}
$$

1. Initialisation
2. 1st iteration
3. 2nd iteration

The Emptiness Test

Algorithm (Emptiness Test)

Input: $G = \langle N, \Sigma, P, S \rangle$

Question: $L(G) = \emptyset$?

Procedure: *mark every* $a \in \Sigma$ *as productive*;

```
repeat
  if there is A → α ∈ P such that all symbols in α productive then
    mark A as productive;
  end;
until no further productive symbols found;
```

Output: *"no" if S productive, otherwise "yes"*

Example

$$
\begin{aligned}
G:\quad S &\rightarrow AB \mid CA \\
A &\rightarrow a \\
B &\rightarrow BC \mid AB \\
C &\rightarrow aB \mid b
\end{aligned}
$$

1. Initialisation
2. 1st iteration
3. 2nd iteration

S productive $\Longrightarrow L(G) \neq \emptyset$

Summary: The Emptiness Problem for Context-Free Languages

Seen:

- Emptiness problem decidable based on productivity of symbols

Summary: The Emptiness Problem for Context-Free Languages

Seen:

- Emptiness problem decidable based on productivity of symbols

Next:

- Closure properties of CFLs

Software Modeling and Verification Chair
RWTH AACHEN UNIVERSITY